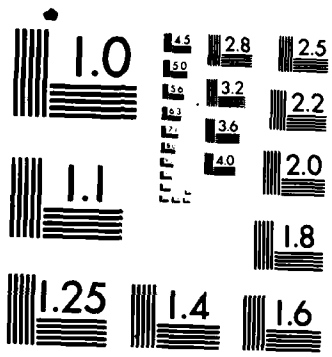
AD-A161 421 REQUEST FOR INSTRUMENTATION(U) CALIFORNIA UNIV BERKELEY 1/1
ELECTRONICS RESEARCH LAB M STONEBRAKER AUG 84
AFOSR-TR-85-0972 AFOSR-83-0349

UNCLASSIFIED F/G 5/2 NL

END



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFOSR-TR- 85-0972
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): Request for Instrumentation

5. TYPE OF REPORT & PERIOD COVERED: Final Scientific Report, 1 Sept. 1983 - Aug. 31, 1984

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): M. Stonebraker

8. CONTRACT OR GRANT NUMBER(s): AFOSR-83-0349

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Electronics Research Laboratory, University of California, Berkeley, CA 94720

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F 2917 A5

11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force office of Scientific Research, Bldg. 410, Bolling Air Force Base, Washington, DC 20332

12. REPORT DATE: Aug 84

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

unlimited

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

DTIC ELECTE NOV 20 1985 S D A

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

INGRES, database, complex objects, extendible DBMS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The authors proposed a research program to develop a generalized database manager to support diverse kinds of data including text, icons, forms, maps and other spatial data. The proposed research also included investigating support for integrated data browsers to allow end-users to query, step through, and update diverse data. Specific topics to be investigated included query language facilities to support text and geometric data, user-defined abstract data types in a DBMS, an ordered relation access method for the text and other ordered data, extended secondary indexes, main memory databases, concurrency control (con't on back)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A161 421

DTIC FILE COPY

05 11 15 128

for data browsers, and an application program interface based on windows.

This paper reports on our progress during the first year of this program. The major advances have been made inthe areas of abstract data types, main memory data bases and extended secondary indexes.

AFOSR-TR-

Final

SCIENTIFIC REPORT

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

REQUEST FOR INSTRUMENTATION

M. Stonebraker
Principle Investigator

September 1, 1983 - August 31, 1984

Approved for public release;
distribution unlimited.

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution /
Availability Codes
Dist
A-1

QUALITY INSPECTED

The goals of this project are an adjunct to those of AFOSR-83-0254. Moreover, the proposal was an attachment to the aforementioned grant. Rather than cosmetically alter that report, it has been included verbatim.

ABSTRACT

The authors proposed a research program to develop a generalized database manager to support diverse kinds of data including text, icons, forms, maps and other spatial data. The proposed research also included investigating support for integrated data browsers to allow end-users to query, step through, and update diverse data. Specific topics to be investigated included query language facilities to support text and geometric data, user-defined abstract data types in a DBMS, an ordered relation access method for text and other ordered data, extended secondary indexes, main memory databases, concurrency control for data browsers, and an application program interface based on windows.

RESEARCH RESULTS DURING THE 1983-1984 YEAR

We have made significant progress in our exploration of abstract data types, main memory data bases and extended secondary indexes during the last year. We comment briefly on significant accomplishments in each of these areas in the following subsections.

Abstract Data Types

One of our goals was to investigate ways of representing more complex objects such as geometric objects, text, maps, etc. in a relational data base system. Our major reseach result involves the possibility of using a query language to represent data types. This research is explored in detail in [STON84], which is included as an appendix to this document. Here we briefly review two previous proposals, then indicate the significance of our current contribution.

Our previous approach to supporting complex objects in a data base system was through abstract data types. We suggested allowing new types of columns to be added to a data base system along with new operators on these columns [STONE83]. For example, a skilled programmer could define polygons as a new data type along with a collection of operators on data of type polygon including an intersection operator (!!). Then, other users could use polygons in the same way they use the built-in types of a data manager (floating point numbers, integers, and character strings). For example, a user could define the following POLYGON relation

create POLYGON (pid = i4, p-desc = polygon)

and then find the polygons overlapping the unit square as follows:

retrieve (POLYGON.all) where POLYGON.p-desc !! "0,0,1,1"

Support for user defined types and new operators has been constructed in about 2500 lines of code for the INGRES relational data base system. Implementation details are addressed in [FOGG82, ONG82], and ADTs run with a modest performance degradation [FOGG82]. Initial suggestions concerning how to integrate new operators into query processing heuristics and access methods are contained in [STON83, ONG84].

The abstract data type approach to complex objects is conceptually clean because no facilities peculiar to a specific kind of data are required. However, it has the disadvantage that one cannot easily "open up" an object and examine its component sub-objects. For example, suppose an airplane wing is defined as an abstract data type composed of a collection of components (e.g. cowlings, engines, etc.). In turn, each component could be composed of sub-components. Then, suppose a user wished to isolate a turbine blade inside a specific engine on a wing. To perform this task he would need two operators, one to "open up" the wing and identify a specific engine and a second to "open up" the engine and

identify the desired turbine blade. Two cascaded operators are an awkward way to search in a complex object for a specific sub-object.

A second approach is to extend a relational data base system with specific facilities for particular complex objects. This is the approach taken in [LOR183] for objects appropriate for CAD applications. It has the advantage that component objects can be addressed but requires special-purpose services from a DBMS.

Our major contribution under the current grant is a third approach which may offer the good features of each of the above proposals. It involves supporting commands in the query language as a data type in a DBMS. In our environment this means that a column of a relation can have values which are one (or more) commands in the data manipulation language QUEL. We explain our proposal using the following example. Suppose a complex object is composed of lines, text and polygons. Each component is described in a separate relation as follows:

```
LINE (Lid, l-desc)
TEXT (Tid, t-desc)
POLYGON (Pid, p-desc)
```

Example inserts into the LINE and POLYGON relation are:

```
append to LINE (Lid = 22,
     description = "(0,0) (14,28)")
append to POLYGON (Pid = 44,
     description = "(1,10) (14,22) (6,19) (12,22)")
```

Then, the object as a whole would be stored in the OBJECT relation:

```
OBJECT (oid, o-desc)
```

The description field in OBJECT would be of type QUEL and contain queries to assemble the pieces of any given object from the other relations. For example, the following insert would make object 6 composed of line 22 and polygon 44.

```
append to OBJECT(
```

```
oid = 6,
o-desc = "retrieve (LINE.all) where LINE.id = 22
      retrieve (POLYGON.all) where pid = 44")
```

We have proposed extensions to QUEL which allow the components of an object to be addressed. For example, one could retrieve all the line descriptions making up object 6 which were of length greater than 10 as follows:

```
range of O is OBJECT
retrieve of (O.o-desc.1-desc) where
      length (O.o-desc.1-desc)>10
```

This notation has many points in common with the data manipulation language GEM [ZANI83], and allows one to conveniently discuss subsets of components of complex objects. In addition we can support clean sharing of lines, text and polygons among multiple composite objects by having the same query in the description field of more than one object, a feature lacking in the proposal of [LORI83].

Main Memory Data Bases

We have spent considerable time exploring the use of large amounts of main memory to speed data base processing. In [DEWI84] we have presented our results. The main contributions are:

1) an investigation of the viability of AVL trees in an environment where most of a relation may be present in main memory

2) an investigation of new algorithms for performing relational joins which can effectively utilize large quantities of main memory

3) an investigation of efficient means of obtaining crash recovery for a data base mostly resident in main memory

Our conclusions are somewhat counterintuitive. For example, if is found that merge-sort [SELI79] is rarely effective as a join tactic in an environment

with much main memory becasuse it is outperformed by several variations of hashing joins. A full discussion of this point and others appears in [DEWI84], which is also included in the appendix.

Extended Secondary Indices

During the past year we have concentrated on designing a secondary indexing structure that would be appropriate for solid geometric objects such as the polygons and rectagons which appear in CAD applications. Conventional spatial indexing schemes (e.g. KDB trees [ROBI81] are only appropriate for point data. Our scheme, R-trees, allows efficient access to solid spatial objects according to their location [GUTT84]. Leaf nodes in an R-tree contain index entries, each consisting of a pointer to a spatial object and a rectangle that bounds the object. Higher nodes contain similar entries, with pointers to lower nodes and rectangles bounding the objects in the lower nodes. This hierarchy of covering rectangles is built and maintained dynamically in a manner similar to a B+tree.

To search for all data overlapping a given rectangle, we examine the root node to find which entries have rectangles overlapping the search area. The corresponding subtrees can have data in the search area, and therefore we apply the search algorithm recursively to each one. In this way we find all qualifying data, but avoid searching parts of the tree corresponding to objects that are far from the search area.

R-trees can be built for any number of dimensions. In addition they are useful for overlapping objects of non-zero size, a characteristic not shared by most multi-dimensional indexing schemes, for example quad trees [FINK74], k-d trees [BENT75], and K-D-B trees [ROBI81].

We have implemented R-trees, and in spatial search tests using VLSI data, only about 150 usec of CPU time was required to find each qualifying item. This indicates that the structure effectively restricts processing to qualifying or

near-qualifying data. Our paper on R-trees [GUTT84] is included in the appendix to this proposal.

REFERENCES

[BENT75] Bentley, J.L., "Multidimensional Binary Search Trees Used for Associative Searching," CACM 18,9 (September 1975), 509-517.

[DEWI84] Dewitt, D., et. al., "Implementation Techniques for Main Memory Database Systems," to appear in Proc. 1984 ACM-SIGMOD Conference on Management of Data, Boston, Mass., June 1984.

[FINK74] Finkel, R.A. and J.L. Bentley, "Quad Trees - A Data Structure for Retrieval on Composite Keys," Acta Informatica 4, (1974), 1-9.

[FOGG82] Fogg, D., "Implementation of Domain Abstraction in the Relatinal Database System, INGRES," Masters Report, EECS Dept, University of California, Berkeley, Sept. 1982.

[GUTT84] Guttman, A., "R-Trees: A Dynamic Index Structure for Spatial Searching," to appear in 1984 ACM-SIGMOD Conference on Management of Data, Boston, Mass., June 1984.

[LORI83] Lorie, R. and W. Plouffe, "Complex Objects and Their Use in Design Transactions," Proc. Engineering Design Applications of ACM-IEEE Data Base Week, San Jose, Ca., May 1983.

[ONG82] Ong, J., "The Design and Implementation of Abstract Data Types in the Relational Database System, INGRES," Masters Report, EECS Dept, University of California, Berkeley, Sept. 1982.

[ONG84] Ong. J., et. al., "Implementation of Data Abstraction in the Relational Database System INGRES," to appear in SIGMOD Record.

[ROBI81] Robinson, J.T., "The K-D-B Tree: A Search Structure for Large Multidimensional Dynamic Indexes," Proc 1981 ACM-SIGMOD Conference, Ann Arbor, Mich., June 1981.

[SELI79] Selinger, PP., et. al., "Access Path Selection in a Relational Data Base Management System," Proc. 1979 ACM-SIGMOD Conference on Management of Data, Boston, Mass., June 1979.

[STON83] Stonebraker, M. et. al., "Application of Abstract Data Types and Abstract Indices to CAD Databases," Proc. Engineering Design Applications of ACM-IEEE Database Week, San Jose, Ca., May 1983.

[STON84] Stonebraker, M., et. al., "QUEL as a Data Type," to appear in Proc. 1984 ACM-SIGMOD Conference on Management of Data, Boston, Mass., June 1984.

[ZANI83] Zaniola, C., "The Database Language GEM," "Proc. 1983 ACM-SIGMOD Conference on Management of Data, San Jose, Ca., May 1983.

PROFESSIONAL PERSONNEL

Michael Stonebraker, PI

Erika Anderson: B.A. (Computer Science), May 1984.

Antonin Gutman: PhD August 1984, "Application of Relational Data Base Systems to CAD Data"

John Woodfill: B.S. (Computer Science), May 1984.

PUBLICATION CITATIONS

David J. Dewitt, Randy H. Katz, Frank Olken, Leonard D. Shapiro, Michael Stonebraker and David Wood, "Implementation Techniques for Main Memory Database Systems," *Proc. 1984 ACM-SIGMOD Conference on Management of Data*, Boston, Mass., June 1984.

Michael Stonebraker, Erika Anderson, Eric Hansen and Brad Rubinstein, "QUEL as a Data Type," *Proc. 1984 ACM-SIGMOD Conference on Management of Data*, Boston, Mass., June 1984.

A. Gutman, "R-Trees: A Dynamic Index Structure for Spatial Searching," *Proc. 1984 ACM-SIGMOD Conference on Management of Data*, Boston, Mass., June 1984.

Michael Stonebraker, "Virtual Memory Transaction Management," *ACM SIGOPS Review*, April 1984.

Michael Stonebraker, John Woodfill and Erika Anderson, "Implementation of Rules in Relational Data Base Systems," *ACM SIGMOD Record*, September 1983.

INTERACTIONS

The first three papers in the Publication Citations section were presented at ACM SIGMOD annual conference.

A presentation on "QUEL as a Data Type" was given at Wang Institute and IBM Federal Systems Division.

END

FILMED

1-86

DTIC